AMENDMENTS TO THE SPECIFICATION

Please amend the paragraph beginning on page 4, line 16 as follows:

In the first stage, Instruction Pointer Generation-1 (IPG-1) 102, a pointer is provided for debug operations 122, such as 'design for testability' (DFT) operations. This instruction pointer path maintains the highest priority for the multiplexer 118. Therefore, this instruction pointer would pre-empt any other instructions received at the multiplexer simultaneously. The next stage in the processor pipeline, instruction pointer generation (IPG) 104, is the stage of the pipeline in which the various re-steers are presented to the multiplexer 118 for priority routing to the processor. All of the re-steer logic paths are fed into the multiplexer 118 with their associated multiplexer priority. The priority is denoted by the order of placement on the multiplexer - the higher the input location, the higher the priority. Also, the re-steer logic components 122,124,126,128,130, 132 have been labeled 1 ~~5~~ 6 to denote their relative multiplexer priority. Basically, the priorities are determined by relative cycle time penalties related to each respective re-steer scenario. The more cycles that will be lost due to a given re-steer, the higher the priority of that re-steer path. For example, a '1' bubble branch re-steer 128 has a one cycle penalty (the one cycle previous to it must be flushed). Therefore, it has a greater priority than a '0' bubble branch re-steer 130, which does not have to flush a cycle. Also, sequential IP 132 is lowest in priority because a simple instruction pointer increment should not be performed unless it is assured that no re-steers need to be performed.

Please amend the paragraph beginning on page 7, line 6 as follows:

Figure 2 illustrates the operation of instruction pointer generation of a typical simultaneous multithreaded processor that utilizes pipelining methodology. In the IPG stage ~~204~~104, a separate multiplexer 218, 220 is utilized for each thread being received by the processor. MUX1 218 is utilized for thread 1, and MUX2 220 is utilized for thread 2. A common multiplexer 246 switches between the two threads, depending upon which one is active. As compared to the single-thread, pipelined processor described in Figure 1, the typical multi-thread version has twice as many inputs for each multiplexer 218,220. For each input necessary for a single-thread processor, there is one input for direct (live) feed 260 from the re-steer logic and one pre-recorded data path (from the associate storage element) 258.

Please amend page 8, line 12 as follows:

Figure 3 provides an illustration of an embodiment of the present invention. As opposed to storing re-steer information in individual storage elements related to each of the re-steer logic components 324,326,328,330,332, in one embodiment of the present invention, the multiplexer 318 or 320 of the inactive thread operates as if the thread was active. However, instead of the output of the multiplexer being fed to the processor pipeline through the common multiplexer 346, the instruction pointer information of the inactive thread is stored in an 'inactive thread' storage element 348 or 350 to be delivered back into the respective multiplexer 318 or 320 when the thread once again becomes active. In an embodiment of the present invention the inactive thread re-steer storage element 348,350 is a flop with an enable. When the inactive thread once

again becomes active, 'inactive thread re-steer' logic 352,354 enables the storage element 348, 350 to release the instruction pointer back into the multiplexer 318, 320. In this embodiment, the inactive thread re-steer is placed at a multiplexer priority between 4 ('1' bubble branch re-steer) and 5 ('0' bubble branch re-steer). The reason for this priority placement is that if the thread at IPG+1 ~~306~~ 106 differs from that at IPG ~~304~~ 104 (changing between active and inactive), '0' bubble branch re-steer 330 and sequential IP 332 would not be utilized (because of the threads changing), and the newly active thread instruction pointer would have been coming from the inactive re-steer storage element (into the multiplexer). Because the '1' bubble branch re-steer 328 was generated a cycle earlier, it has a higher priority than the inactive thread re-steer 352,354. If the thread at IPG ~~304~~ 104 is the same as the thread at IPG+1 ~~306~~106, the inactive thread re-steer storage element 348,350 would not be enabled, and both '0' bubble branch re-steer 330 and sequential IP 332 would be possible.